VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Rozpoznávání anomálií v obličeji řidiče

Anomaly Detection of Driver's Face

2020 Pavel Mandrla

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Pavel Mandrla**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Rozpoznávání anomálií v obličeji řidiče
Anomaly Detection of Driver's Face

Jazyk vypracování: čeština

Zásady pro vypracování:

Rozpoznání anomálií ve tváři v obrazech je v posledních letech hodně rozvíjené téma. Aplikace tohoto druhu může být použita například v oblasti samořiditelných vozidel.

1. Popište základní pojmy a metody v oblasti rozpoznávání anomálií v obrazech. Zaměřte se především na lidské tváře.
2. Seznamte se s volně dostupnými knihovnami a popište jaké možnosti nabízí v této oblasti (například OpenCV, Dlib, TensorFlow).
3. S pomocí knihoven vytvořte vybraný detektor (rozpoznávač) anomálií v obličeji řidiče.
4. Experimentálně ověřte funkčnost, přesnost a rychlost navrženého řešení na dostupných datasetech.
5. Své závěry řádně zdokumentujte v textu práce.

Seznam doporučené odborné literatury:

[1] Perera, Pramuditha & M. Patel, Vishal. Learning Deep Features for One-Class Classification. IEEE Transactions on Image Processing. PP. 10.1109/TIP.2019.2917862. (2018)
[2] R. Chalapathy, A. K. Menon, S. Chawla, "Anomaly detection using one-class neural networks", arXiv: 1802.06360, Aug. 2018

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Radovan Fusek, Ph.D.**

Datum zadání: 01.09.2019
Datum odevzdání: 30.04.2020

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal

V Ostravě 14. května 2020 .....

Rád bych poděkoval vedoucímu bakalářské práce Ing. Radovanu Fuskovi, Ph.D. za odbornou pomoc a cenné rady, které mi při psaní této bakalářské práce poskytl.

Abstrakt

Tato bakalářská práce se zabývá detekcí anomálií v obličeji řidiče. V teoretické části se věnuje vysvětlení základních pojmů v této oblasti, metodám, které se dají pro detekci anomálií použít, způsobům zpracování vstupního obrazu pro zlepšení vlastností detekce a dostupným knihovnám, které jsou použitelné pro implementaci detektoru. Dále se zabývá tím, jaká data, kromě obličeje řidiče, by mohl detektor pro klasifikaci použít a zlepšit tak svou přesnost.

Praktická část této práce se zabývá implementací detektoru anomálií a následným otestováním jeho vlastností na testovacích datech.

Klíčová slova: strojové vidění, detekce anomálií, SVM

Abstract

This bachelor thesis is concerned with detection of anomalies in the face of a driver. The first part describes the basic concepts in this area, methods, which can be used for anomally detection, processing of the input image used for better performance and accuracy of the classifier and also the available libraries and frameworks, which can help with the implementation of an anomaly detector. It also proposes which information, other than the drivers face, can be used by the detector to clasify anomalies and improve its accuracy.

The second part describes the implementation of a facial anomally detector and testing its capabilities on testing data.

Keywords: machine vision, anomaly detection, SVM

Obsah

Seznam použitých zkratek a symbolů

ABS	–	Anti Blocking System
ACC	–	Accuracy
DNN	–	Deep neural network
DOC	–	Deep One-Class Classification
HOG	–	Histogram of Oriented Gradients
LBP	–	Local Binary Patterns
LBPH	–	Local Binary Patterns Histogram
NPV	–	Negative Predictive Value
OC-CNN	–	One-class convolutional neural network
OC-SVM	–	One-class support vector machine
PPV	–	Positive Predictive Value
SVM	–	Support Vector Machine
EEG	–	Elektroencefalograf
EOG	–	Elektrookulograf
EMG	–	Elektromyograf

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Automobilky i jejich zákazníci kladou v současné době stále větší důraz na bezpečnost automobilů. Bezpečnost vozidla již dávno nezajišťují pouze pasivní bezpečnostní prvky jako bezpečnostní pásy, airbagy či deformační zóny karoserie, které slouží k záchraně životů při autonehodě, ale i prvky aktivní bezpečnosti, které se snaží nehodám předcházet a zabránit jim.

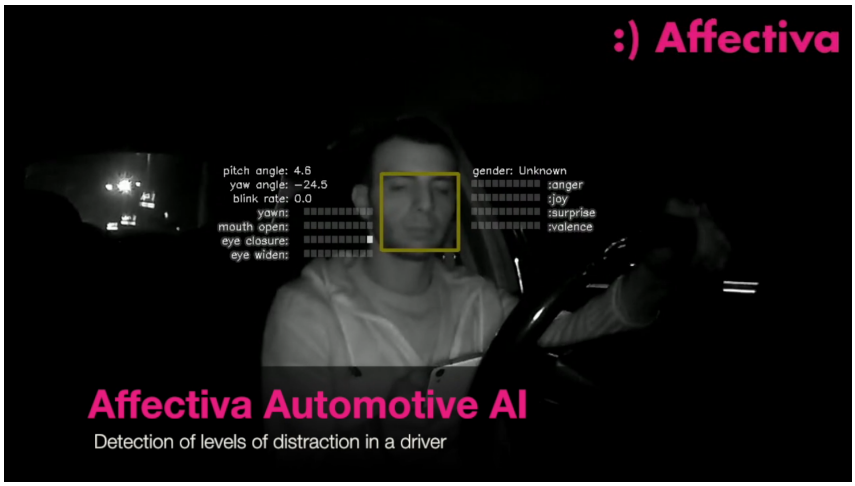
K prvkům aktivní bezpečnosti patří kvalitní brzdy, přesné řízení automobilu, dobrý výhled z vozidla, ale i více high-tech řešení, jako je třeba protiblokovací systém ABS. S vylepšujícími se technologiemi prvků aktivní bezpečnosti přibývá a už k nim nepatří pouze systémy pomáhající řidiči zvládnout řízení vozidla.

Jednou z nejčastějších příčin autonehod je v dnešní době nepozornost řidiče při řízení vozidla, jelikož se řidiči při řízení často věnují jiným činnostem. Důležitým faktorem je také únava, která na řidiče může padnout například během dlouhých jízd nebo pokud řidič řídí v noci a není na to zvyklý. Řidiči často ignorují doporučení dávat si během jízdy odpočinkové pauzy, což nevyhnutelně vede k nebezpečným situacím na vozovce. Z těchto důvodů se nově k prvkům aktivní bezpečnosti řadí i technologie kontrolující samotného řidiče a jeho jízdu. K nim patří například systémy kontrolující, zda řidič nevyjíždí při jízdě z pruhu nebo systémy kontroly bdělosti řidiče.

Systémy rozpoznávající anomálie v chování řidiče by jej mohly v situaci, kdy se nevěnuje řízení, upozornit a případně i dočasně převzít kontrolu nad vozidlem, aby zůstala zachována bezpečnost provozu. Staly by se tak velice cennými nástroji, které by chránily nejen život samotného řidiče, ale i životy ostatních účastníků provozu.

Teoretická část této práce se věnuje metodám, které mohou být použity při vytváření detektoru anomálií v obličeji řidiče, a také způsobům, jak zlepšit přesnost a rychlost detekce detektoru.

Praktická část práce se zabývá implementací detektorů anomálií založených na metodě podpůrných vektorů a příznacích HOG a LBPH. Následně je testována jejich funkčnost a výkonnost.



Obrázek 1: Ilustrační snímek (Převzato z [1])

2 Detekce anomálií

Detekce anomálií se zabývá problémem hledání vzorů v datech, které se chovají jinak, než se očekává. Takový prvek se nazývá anomálie. Anomálie lze rozdělit do tří kategorií [2].

Bodové anomálie

Jde o bod v datech, který se jasně odlišuje od všech ostatních a dá se proto snadno klasifikovat jako anomálie. Je to nejjednodušší typ anomálie.

Příkladem bodové anomálie je situace, kdy jsou normálními instancemi všechna čísla od nuly do pětky. Hodnota 23 mezi tyto hodnoty jasně nepatří a nepochybně se jedná o anomálii.

Kontextuální anomálie

Pokud jsou data anomální pouze v určitém kontextu, avšak jindy se řadí mezi normální data, pak se taková anomálie nazývá kontextuální.

U detekce anomálií v obličeji řidiče by mohla být kontextuální anomálií situace, kdy řidič přivírá oči. Pokud tak činí z důvodu, že jede proti slunci, které mu svítí do očí, je taková situace normální. Je-li ale ospalý a za volantem usíná, jedná se o anomálii.

Kolektivní anomálie

Pokud je skupina dat anomální vůči zbytku datasetu, ale jednotlivé body, které skupinu tvoří, anomální nejsou, pak se tyto body nazývají kolektivní anomálie. Detektor by je měl jako anomální označit až ve chvíli, kdy se začínají objevovat pospolu v příliš velkých počtech.

I tady by příkladem mohly být zavřené oči řidiče. Pokud má řidič zavřené oči jenom na jednom snímku, pak to anomálie není, protože byl snímek nejspíše vyfocen zrovna ve chvíli, kdy řidič mrknul. Pokud má však zavřené oči delší dobu, už je to nebezpečné a mělo by to být klasifikováno jako anomálie.

Detekování anomálií v obličeji řidiče je důležité pro zvýšení bezpečnosti silničního provozu. Nevěnování se řízení vozidla se ve statistikách počtu zavinění dopravních nehod dlouhodobě pohybuje na předních příčkách. V roce 2018 se stalo 8 709 dopravních nehod právě z tohoto důvodu a podle Policie České republiky se jedná o nejčastější příčinu vzniku dopravních nehod zaviněných řidiči automobilů v tomto roce [3].

Mezi hlavní důvody ztráty pozornosti řidiče patří používání mobilního telefonu, únava, mikrospánek, ladění rádia, nastavování GPS navigace, kouření, konzumace potravin nebo komunikace se spolujezdci. Toto jsou anomálie v řidičově chování, které by měl detektor detekovat a řidiče v takovou chvíli upozornit, aby se věnoval řízení, nebo zastavil vozidlo.

Mnohé automobilky již dnes do svých automobilů montují systémy, které sledují řidičovu jízdu a pokud detekují, že řidič začíná ztrácet koncentraci, upozorní jej, že by měl zastavit

vozidlo a udělat si přestávku. Mezi tyto systémy patří například Driver Alert Control (DAC) [4] používaný společností Volvo, Driver Alert System [5] používaný v automobilech koncernu Volkswagen nebo Attention Assist [6] od firmy Mercedes Benz. Tyto systémy únavu detekují z dat získaných z automobilu. Analyzují dráhu jízdy automobilu, pohyby volantem, tlak na plynový pedál, změny stylu jízdy či pozici automobilu mezi podélným značením na vozovce. Podle těchto informací rozhodují, zda se řidič řízení aktivně věnuje. Jsou však uzpůsobeny k jízdě po hlavních silnicích a zapínají se až po dosažení určité rychlosti.

Existují také metody, které při sledování únavy řidiče využívají behaviorální a fyziologická data. Mezi behaviorální data se řadí především informace, které jsou získané z kamery umístěné v kabině automobilu tak, aby snímala řidičův obličej. Fyziologická data jsou získávána z řidičových biologických hodnot [7]. Do této kategorie spadají například elektrokardiografie měřící aktivitu srdce, elektrookulografie měřící aktivitu očí či elektroencefalografie zaznamenávající elektrický potenciál mozku.

Chellappa, Joshi a Bharadwaj ve svém článku [8] popisují detektor únavy řidiče, který o únavě rozhoduje právě z behaviorálních a fyziologických dat. Kromě snímání řidiče kamerou navíc získávali data z teploměru měřícího teplotu řidičova těla a z pulzního oxymetru, který měřil řidičův pulz. V získaných kamerových snímcích detekují mrkání a zívání. Jejich systém byl navržen tak, aby detekoval tři úrovně únavy od lehké, kdy byla řidiči pouze navrhnuta přestávka na kávu, po vážnou únavu, při které systém spustil zvukové varování, které mělo řidiče přimět k tomu, aby zastavil vozidlo.

2.1 Metody detekce anomálií

Metody detekce anomálií se dají rozdělit podle typu učení, které je využito pro sestavení modelu, na základě kterého jsou vstupní data klasifikována jako anomální, nebo normální. Tyto typy učení se dělí podle toho, jak jsou instance v datasetu, ze kterého se detektor učí, označkovány. Instance jsou při detekci anomálií rozděleny do dvou tříd, a to anomální a normální. Podle výskytu těchto tříd v datasetu se rozlišují následující typy [2].

Detekce anomálií využívající učení s učitelem (Supervised Anomaly Detection)

Metody detekce anomálií založené na tomto přístupu počítají s tím, že všechny instance v datasetu, ze kterého se klasifikátor učí, mají u sebe informaci, zda se jedná o anomálii nebo ne. Značkování těchto dat se většinou musí provádět ručně, tudíž získání takového datasetu je velice náročné. Navíc tento dataset musí počítat se všemi možnými anomáliemi, které se mohou vyskytnout, což problém získávání dat dělá ještě složitějším.

Jelikož v lidských obličejích a jejich výrazech existuje velké množství variací, je sestavení takového datasetu pro trénování klasifkátoru velice náročný úkol a použití těchto metod pro detekci anomálií v obličeji je tedy nesmírně obtížné.

Detekce anomálií využívající kombinaci učení s učitelem a bez učitele (Semisupervised Anomaly Detection)

U tohoto typu obsahuje dataset používaný pro trénování modelu pouze instance normální třídy. Cokoliv, co se nepodobá instancím v datasetu, je detektorem klasifikováno jako anomálie. Získání takovéhoto datasetu je mnohem jednodušší, než v případě učení s učitelem.

Detekce anomálií využívající učení bez učitele (Unsupervised Anomaly Detection)

U učení bez učitele nejsou data v datasetu nijak označkována, tudíž není jasné, co je anomální a co ne. Algoritmy používající tento přístup předpokládají, že anomální data se v datasetu nevyskytují tak frekventovaně, jako ty normální a výrazně se od nich liší [9].

Zbytek této kapitoly se věnuje některým algoritmům pro detekci anomálií.

2.1.1 One-class SVM

Jedním z klasifikačních algoritmů použitelných pro detekci anomálií je One-class SVM. Jedná se o speciální případ metody podpůrných vektorů (anglicky Support Vector Machine).

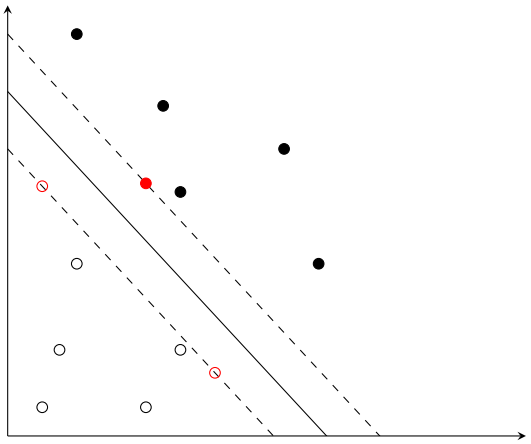
Metoda podpůrných vektorů se snaží od sebe oddělit body dvou tříd ve vektorovém prostoru pomocí optimální nadroviny. Následně tuto nadrovinu používá jako rozhodovací hranici, podle které přiřazuje instance v příznakovém prostoru jednotlivým třídám. Instance ležící na jedné straně této nadroviny, patří k jedné třídě, zatímco instance nacházející se na straně opačné náleží k třídě druhé. Nadrovina se dá popsat rovnicí 1.

$$\vec{w}x + r = 0 \tag{1}$$

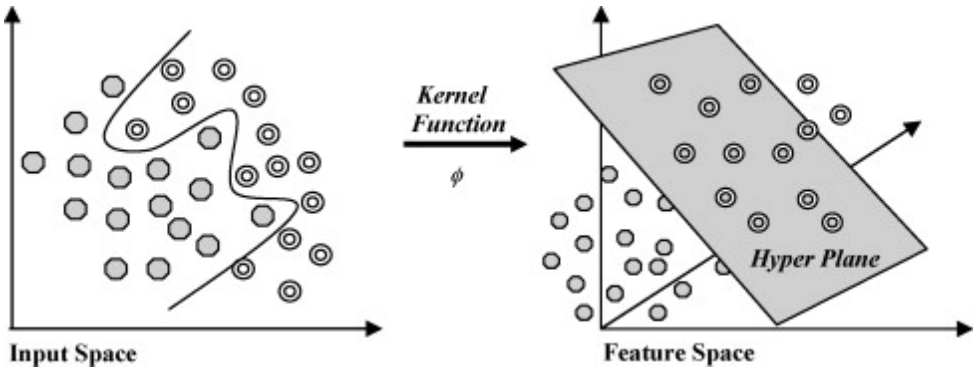
$\vec{w}$ je normálový vektor nadroviny a r je její posun v prostoru. $\frac{r}{||\vec{w}||}$ se rovná vzdálenosti nadroviny od počátku příznakového prostoru. Pokud by se r rovnalo nule, procházela by nadrovina počátkem příznakového prostoru.

Existuje mnoho způsobů, jakými by se dala nadrovina do příznakového prostoru vložit. Aby došlo k co největšímu oddělení instancí obou tříd a pozice nadroviny tak byla optimální, umisťuje ji algoritmus mezi body obou tříd tak, aby vzdálenost k nejbližším bodům obou tříd byla co největší [10]. Tyto nejbližší body se nazývají podpůrné vektory. Jejich odstranění z datasetu při trénování by vedlo k jinému umístění optimální nadroviny. Nadrovina tedy prochází nejširším možným pásmem v příznakovém prostoru, v němž se nenachází žádné instance jednotlivých tříd. Podpůrné vektory se nacházejí na okraji tohoto pásma. Příklad vložení optimální nadroviny do příznakového prostoru je vyobrazen na obrázku 2.

Není těžké představit si situaci, kdy se instance obou tříd nedají lineárně oddělit a nelze tak jednoznačně určit polohu nadroviny. V takovém případě se dá použít nelineární SVM. Příznakové vektory jsou promítnuty pomocí jádrové funkce ϕ do vícedimenzionálního prostoru, ve kterém již lze nadrovinu pro separaci obou tříd určit. Tento postup je znázorněn na obrázku 3.



Obrázek 2: Oddělení bodů dvou tříd pomocí nadroviny. Podpůrné vektory jsou vyznačeny červenou barvou.



Obrázek 3: Použití jádrové funkce pro trasformaci příznakového vektoru do vyšší dimenze, kde se body dají oddělit pomocí nadroviny [11]

Existuje několik různých druhů jádrových funkcí, které se dají pro promítnutí příznakových vektorů do vícedimenzionálního prostoru použít. Patří k nim například lineární, polynomiální, exponenciální či radiální bázová jádrová funkce. Při použití různých jádrových funkcí bude výsledné rozložení instancí ve vícedimenzionálním prostoru odlišné. Vhodným zvolením jádrové funkce se dá dosáhnout větší separace obou tříd.

Metoda podpůrných vektorů však pracuje se dvěma třídami instancí, což znamená, že patří mezi klasifikační algoritmy využívající učení s učitelem. Použití této metody pro detekci anomálií v obličeji by proto bylo velice náročné. Mnohem více se pro tento účel hodí One-class SVM, což je speciální případ SVM, který nadrovinu do vektorového prostoru umisťuje pouze na základě bodů jedné třídy. Spadá tedy pod klasifikační algoritmy využívající kombinaci učení s učitelem a bez učitele. Tento algoritmus poprvé představili ve svém článku Schölkopf, Williamson, Smola, Shawe-Taylor a Platt [12].

Po transformaci jednotlivých instancí v datasetu pomocí jádrové funkce, je počátek transformovaného vektorového prostoru brán jako instance anomální třídy. Následně jsou stejně jako

u normálních SVM odděleny body obou tříd od sebe optimální nadrovinou [13]. Pro umístění této nadroviny do vektorového prostoru je nutné vyřešit následující optimalizační problém.

$$min_{w,\xi_i,\rho}\frac{1}{2}||w||^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \tag{2}$$

n je počet příznakových vektorů ξ trénovacího datasetu a ν je parametr, který udává, jak moc je nadrovina citlivá na falešně pozitivní instance v datasetu. Tento optimalizační problém se dá vyřešit pomocí Lagrangeových multiplikátorů. Anomálie následně mohou být detekovány pomocí klasifikační funkce 3. Instance normální třídy budou ohodnoceny hodnotou 1 a instance třídy anomální hodnotou -1.

$$f(x) = sgn((w \cdot \phi(x_i)) - \rho) \tag{3}$$

Rovnice 2 a 3 byly převzaty z [14].

2.1.2 Algoritmus k-nejbližších sousedů

Algoritmus k-nejbližších sousedů je klasifikační algoritmus, který spadá mezi algoritmy využívající učení s učitelem. V trénovací fázi si algoritmus uloží jednotlivé příznakové vektory a jejich třídy jako body ve vektorovém prostoru. Nevýhodou tohoto přístupu je vysoký nárok na paměť.

Při následné klasifikaci si v uložených datech najde k bodů, které se ve vektorovém prostoru nacházejí nejblíže ke klasifikované instanci. Následně zjistí, která třída se mezi nalezenými body vyskytuje nejčastěji a k této třídě přiřadí klasifikovanou instanci [15].

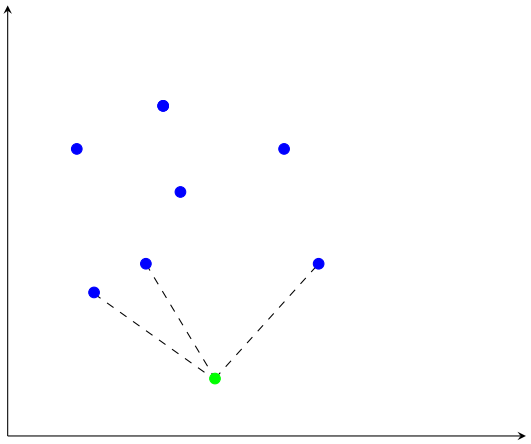
Hodnota k může být jakékoliv celé číslo z intervalu $< 1, n >$, kde n je velikost trénovacího datasetu. Jeho volba však může výrazně ovlivnit výsledky detekční funkce.

Pokud je hodnota $k = 1$, bude klasifikační funkce rychlejší než u větších hodnot k, ale výsledek bude podléhat šumu v datech. Pokud by se v testovacím datasetu nacházely špatně klasifikované instance, mohly by snadno ovlivnit výsledek klasifikace.

Pokud bude k vysoké, bude klasifikační funkce pomalejší a při zvolení příliš vysokého k začne její výsledek ztrácet význam. V případě, že by bylo k tak velké jako je velikost trénovacího datasetu, klasifikovala by funkce každou instanci jako nejčastěji se vyskytující třídu v trénovacím datasetu.

Tento algoritmus lze použít i jako klasifikátor využívající kombinaci učení s učitelem a bez něj a tudíž se dá použít i pro detekci anomálií v obličeji. Jelikož se však v takovém případě v datasetu nachází pouze instance jediné třídy, musí klasifikace instancí probíhat jiným způsobem.

Pro klasifikování třídy instance se již nepoužívají typy tříd k-nejbližších sousedů, nýbrž součet vzdáleností mezi každým z k-nejbližších sousedů a klasifikovanou instancí. Pokud je tento součet vyšší, než je stanovená hranice, nenachází se instance dostatečně blízko skupině neanomálních



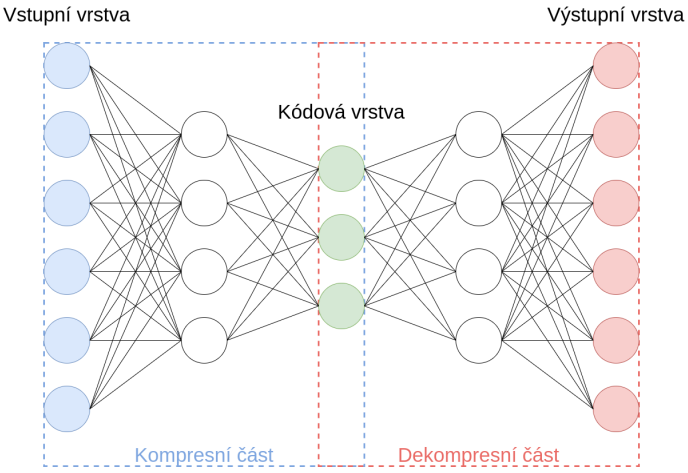
Obrázek 4: Vzdálenosti nejbližších sousedů pro $k = 3$

bodů a jedná se o anomálii. Pokud je tento součet menší, je instance klasifikovaná jako normální [16].

2.1.3 Využití neuronových sítí pro detekci anomálií

Neuronové sítě se v oblasti vícetřídní klasifikace velmi dobře osvědčily. Z tohoto důvodu se také zvýšil objem výzkumu v oblasti využití neuronových sítí pro jednotřídní klasifikaci.

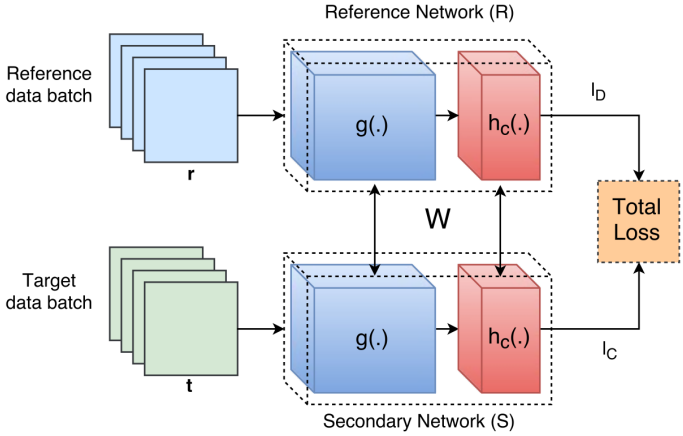
Tato kapitola se věnuje použití neuronových sítí pro detekci anomálií. Tyto sítě se zpravidla skládají ze dvou částí, kdy první část zajiťuje zmenšení dimenzionality vstupu a sestavení příznakového vektoru, zatímco druhá část se zabývá klasifikací instance na základě dat získaných první částí. V této kapitole je popsáno využití autoenkodéru, nebo hluboké neuronové sítě pro sestavení příznakového vektoru a použití jednotřídních hlubokých neuronových sítí pro detekci anomálií.



Obrázek 5: Struktura autoenkodéru

Autoenkodér je neuronová síť, která se snaží o zkopírování vstupních hodnot na svůj výstup [17]. Hodnoty ze vstupní vrstvy ale předtím musí projít kompresní části autoenkodéru. Jejím úkolem je snížit dimenzionalitu vstupu a výsledek předat na své poslední vrstvě, která obsahuje méně neuronů, než vrstva vstupní, části dekompresní. Tato vrstva se nazývá kód a obsahuje ménědimenzionální reprezentaci vstupních dat. Dekompresní část této sítě tyto informace vezme a snaží se z nich co nejlépe zreplikovat data, která dostala kompresní část na svém vstupu.

Struktura autoenkodéru tedy připomíná přesýpací hodiny. Jelikož informace musí v síti projít úzkým hrdlem, které představuje kódová vrstva, musí kompresní část prioritizovat důležité informace, které vstupní data co nejlépe popisují, a ze kterých dokáže dekompresní část vstup co nejlépe napodobit. Z tohoto důvodu jsou hodnoty v kódové vrstvě vhodné pro použití jako hodnoty příznakového vektoru. Po dokončení trénování autoenkodéru lze její kompresní část použít pro extrakci příznaků.



Obrázek 6: Struktura DOC při trénování [18]

Dalšímu způsobu extrakce příznaků pomocí neuronové sítě se ve svém článku věnují Perera a Vishal [18]. Jako základ jejich extraktoru příznaků použili již natrénovanou konvoluční neuronovou síť AlexNet sloužící k vícetřídní klasifikaci. Z této sítě použili její část, která slouží k extrakci příznaků z obrazu předtím, než je klasifikován a upravili její hodnoty tak, aby výsledný příznakový vektor byl co nejvhodnější pro jednotřídní klasifikaci. Tuto metodu nazývají Deep One-Class Classification (DOC). Při dolaďování enkodéru se snaží, aby byly instance odlišných tříd od sebe co nejvzdálenější, a aby se instance stejných tříd shlukovaly. Z tohoto důvodu se zaměřují na následující dvě vlastnosti výsledných příznakových vektorů:

Kompaktnost

Kompaktnost říká, jak blízko u sebe se ve vektorovém prostoru nachází jednotlivé instance třídy, kterou detekor označuje jako normální. Je žádoucí, aby se podobné instance shlukovaly k sobě a bylo je tak jednodušší přiřadit k dané třídě.

Popisnost

Popisnost vyjadřuje schopnost naučených příznaků popsat různá data, která náleží i jiným třídám, než je ta třída, která je použita pro trénování klasifikační funkce.

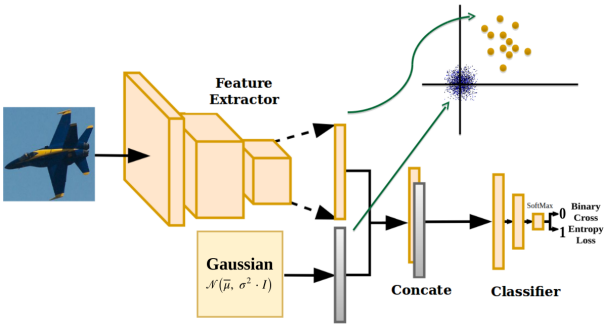
V trénovací fázi vytvořili dvě kopie extraktoru příznaků sítě AlexNet, které nazývají referenční a sekundární. Jako vstup pro referenční síť použili dataset, který byl použit pro trénování sítě AlexNet. Tato síť sloužila pro získání hodnoty popisnosti. Vstupem pro sekundární síť byla data třídy, pro kterou byl trénovaný jednotřídní klasifikátor. Na výstupu sekundární sítě měřili kompaktnost výsledných instancí.

Při trénování upravovali váhy neuronů posledních čtyř vrstev tak, aby byly v obou sítích identické a přitom minimalizovali hodnotu funkce 4.

$$l_{(r,t)} = l_D(r|W) + \lambda l_C(t|W) \tag{4}$$

$l_D(r|W)$ vyjadřuje hodnotu popisnosti neuronové sítě pro referenční data, $l_C(r|W)$ vyjadřuje hodnotu kompaktnosti neuronové sítě pro data cílové třídy a λ je konstanta, která určuje, jestli více záleží na popisnosti, nebo kompaktnosti výsledných dat.

Po dokončení trénování této sítě může být sekundární síť používána pro snížení dimenzionality příznakového vektoru, který je následně klasifikován.



Obrázek 7: Princip trénování OC-CCN [19]

Pro samotnou detekci anomálií se používají jednotřídní hluboké neuronové sítě. Příkladem takovéto sítě může být síť, kterou představili Poojan a Oza ve svém článku [19]. Autoři se při návrhu této sítě inspirovali v již existujících přístupech k detekci anomálií. Podobně jako Shölkopf a spol. [12] používají počátek příznakového prostoru jako instanci anomální třídy.

Při trénování této klasifikační sítě používali příznakové vektory obsahující normální data vytvořená pomocí sítě pro extrakci příznaků a příznakové vektory obsahující data vytvořená pomocí Gaussova šumu shlukující se kolem počátku příznakového prostoru a reprezentující anomální instance. Na základě těchto dat si neuronová síť vytvořila model, podle kterého detekuje anomálie

Jednotřídními neuronovými sítěmi se ve svém článku zabývají i Chalapathy, Menon a Chawla [20]. Jako vstup používá jejich neuronová síť autoenkodér pro snížení dimenzionality a získání

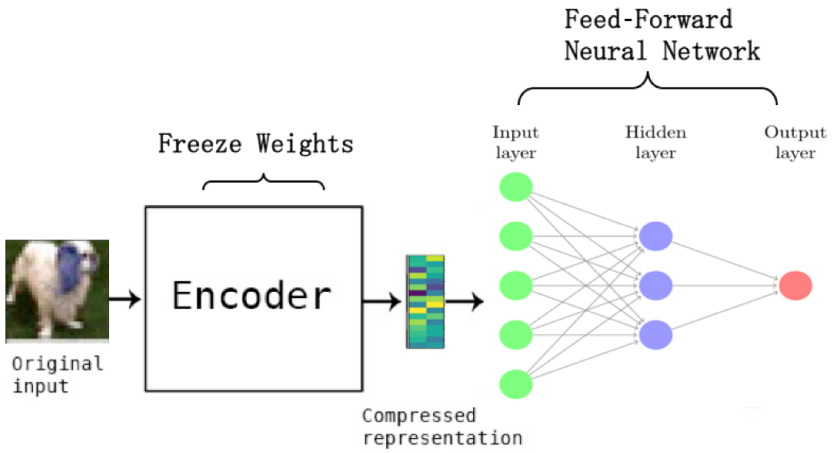
vhodných příznaků. Samotná klasifikační síť je tvořená jednou skrytou vrstvou, jejíž neurony používají lineární nebo sigmoidální aktivační funkci $g(.)$, a výstupní vrstvou, která je tvořena jedním neuronem.

Jejich neuronová síť se inspiruje v OC-SVM. Optimalizační problém jejich sítě, je totiž pozměněným optimalizačním problémem OC-SVM, který ve své práci definují takto:

$$min_{w,r}\frac{1}{2}||w||_2^2 + \frac{1}{\nu N}\sum_{n=1}^{N}\max(0, r - \langle w, \phi(X_n)\rangle) - r \tag{5}$$

Největší změnou, kterou v tomto optimalizačním problému udělali, je záměna skalárního součinu $\langle w, \phi(X_n)\rangle$ za skalární součin $\langle w, g(VX_n)\rangle$, kde V je matice obsahující váhy jednotlivých synapsí mezi vstupní a skrytou vrstvou. Výsledný optimalizační problém jejich neuronové sítě vypadá následovně:

$$\min_{w,V,r}\frac{1}{2}||w||_2^2 + \frac{1}{2}||V||_F^2 + \frac{1}{\nu N}\sum_{n=1}^{N}\max(0, r - \langle w, g(VX_n)\rangle) - r \tag{6}$$



Obrázek 8: Struktura OC-CNN [20]

3 Sestavení příznakového vektoru

Ze vstupního obrázku, pro který rozhodujeme, zda je anomální či ne, musíme před klasifikací vytvořit příznakový vektor. Jde o vektor čísel, nebo znaků předem stanovené abecedy, jehož účelem je reprezentovat, co se v obrázku děje. Tento vektor následně slouží jako vstup pro klasifikátor. Jednotlivé příznakové vektory můžeme brát jako body v příznakovém prostoru, což je vektorový prostor, který má stejný počet dimenzí jako příznakový vektor.

Jako nejzákladnější příznakový vektor je možné vzít vektor, jehož hodnoty odpovídají hodnotám pixelů ve vstupním obrázku. Za předpokladu, že vstupním obrázkem je snímek barevného videa Full HD kvality, by měl tento příznakový vektor 6 220 800 dimenzí. Přitom velká část informací obsažených v tomto příznakovém vektoru je nepotřebná.

V takovém případě dochází k takzvanému prokletí dimenzionality. To znamená, že informace, které jsou pro detektor užitečné, jsou schovány mezi zbytečnými daty a jejich hustota se snižuje.

S větším počtem dimenzí narůstá obtížnost učení detektoru. Aby detektor dosáhl požadované přesnosti, potřebuje více vzorků, ze kterých se bude trénovat, a větší výpočetní výkon stroje, na kterém poběží.

Prokletí dimenzionality se však dá předejít odstraněním přebytečných informací a projekcí příznakového vektoru do ménědimenzionálního příznakového prostoru. Metody redukce dimenizionality se dělí do následujících dvou kategorií [21].

Selekce příznaků

Při selekci příznaků jsou z původního příznakového vektoru vybrány pouze ty příznaky, které jsou pro detektor relevantní.

Příkladem selekce příznaků může být oříznutí vstupního obrázku tak, aby obsahoval pouze obličej řidiče, ve kterém mají být detekovány anomálie.

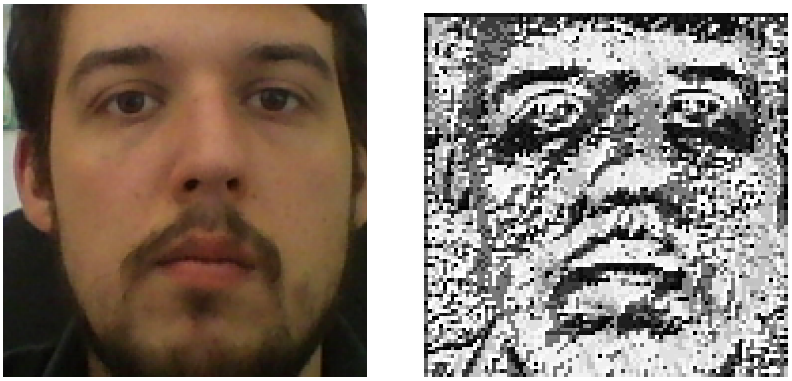
Extrakce příznaků

Metody extrakce příznaků berou hodnoty původního příznakového vektoru a vypočítávají z nich nové hodnoty, které ukládají do výsledného vektoru.

Jednoduchým příkladem extrakce příznaků by mohlo být například převedení vstupního tříkanálového barevného obrázku na černobílý, kde se hodnota každého černobílého pixelu počítá pomocí rovnice $V = 0.3R + 0.59G + 0.11B$.

Použitím komplexnějších metod se dá dosáhnout mnohem větší redukce dimenze. Pro tento účel je možné využít například neuronové sítě popsané v kapitole 2.1.3. Dimenzionalita příznakového vektoru se také dá zmenšit pomocí vizuálních deskriptorů, které slouží k popisu důležitých příznaků nacházejících se v obrázku. Vizuální deskriptory mohou popisovat například hrany, jednoduché tvary či textury. Zbývající část této kapitoly se zabývá několika metodami extrakce příznaků, které se používají pro sestavení příznakového vektoru.

3.1 Histogram lokálních binárních vzorů



(a) Originální obrázek (b) Obrázek s hodnotami LBP

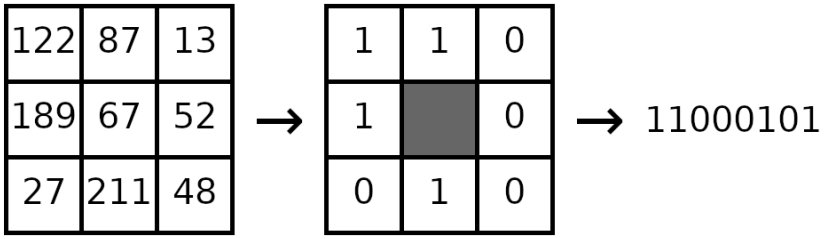
Obrázek 9: Obrázek obličeje, pro jehož pixely byly vypočítány jejich LBP hodnoty

Vizuální deskriptor LBP (lokální binární vzory) byl vytvořen v roce 1994 pro popis vlastností textur v obrázku [22]. Vychází z takzvané Texture Unit, kterou představili o 4 roky dříve Dong-Chen He a Li Wang. Jelikož LBP hodnoty popisují relativní změny v jasu mezi pixely, je tento deskriptor invariantní vůči změnám jasu v obraze [23]. Výsledný příznakový vektor podexponovaného snímku tedy bude stále podobný příznakovému vektoru správně vyexponovaného snímku.

LBP hodnota se počítá z hodnot pixelů ležících na kružnici, jejímž středem je pixel, kterému výsledná hodnota náleží. Ve formě, v jaké byl tento deskriptor původně představen, je poloměr této kružnice $r = 1$. Hodnoty LBP vstupního jednokanálového obrázku se v tomto případě počítají následujícím způsobem [22, 23].

Hodnota pixelu, pro který se LBP hodnota počítá, se porovná s hodnotami všech osmi pixelů v jeho nejbližším okolí. Je-li hodnota sousedního pixelu nižší nebo rovna hodnotě hlavního pixelu, je tento soused ohodnocen hodnotou 1. Jinak dostane hodnotu 0.

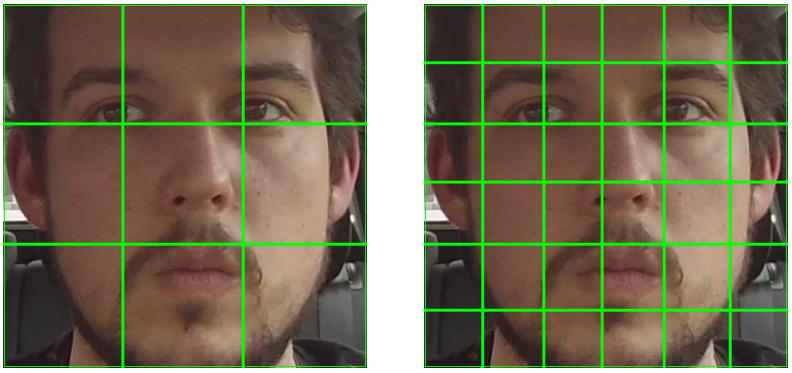
Následně je ohodnocení sousedů převedeno na osmibitovou číselnou hodnotu tak, že ohodnocení každého souseda je použito jako hodnota jednoho bitu osmibitového čísla. Tento postup je znázorněný na obrázku 10.



Obrázek 10: Znázornění výpočtu hodnoty LBP pro jeden pixel

Jelikož jsou hodnoty pixelů vstupního obrázku také vyjádřeny pomocí osmibitového čísla, počet dimenzí příznakového vektoru se nezměnil. Redukce dimenzionality je dosaženo použitím histogramu lokálních binárních vzorů (LBPH). Ten je vytvořen spočítáním výskytů jednotlivých LBP hodnot v obrázku.

Pokud je však celý obličej popsán jediným histogramem, dochází už ke ztrátě příliš mnoha informací a velkému zobecnění řidičova výrazu, což přesnost klasifikační funkce ještě zhorší. Z tohoto důvodu je vstupní obrázek rozdělen na buňky, kde každá buňka je popsána vlastním histogramem. Histogramy všech buněk jsou následně zřetězeny do jednoho velkého histogramu, který se stane výsledným příznakovým vektorem.

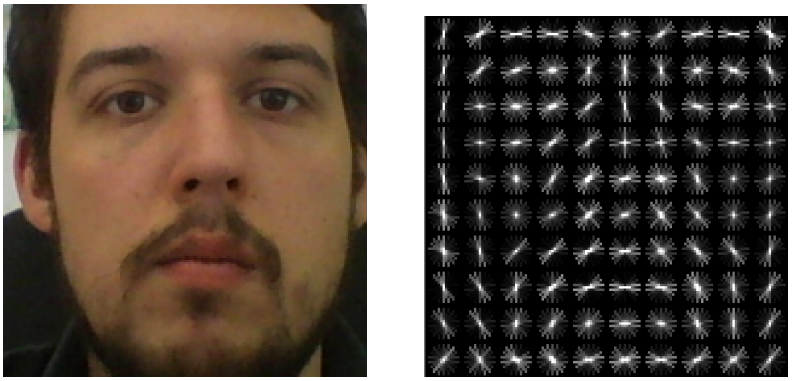


(a) Buňky velikosti 32 pixelů (b) Buňky velikosti 16 pixelů

Obrázek 11: Rozdělení obličeje na buňky

Pro vylepšení vlastností příznaku LBPH mohou být LBP hodnoty před výpočtem histogramu upraveny, aby byly invariantní vůči rotaci [24]. V takovém případě se LBP hodnota upraví tak, aby pořadí jednotlivých bitů za sebou zůstalo stejné, ale výsledná hodnota byla co nejmenší.

3.2 Histogram orientovaných gradientů



(a) Originální obrázek (b) Vizualizace příznaků HOG

Obrázek 12: Vizualizace deskriptoru HOG aplikovaného na obrázek lidského obličeje

Histogram orientovaných gradientů (HOG) je vizuální deskriptor, který byl vyvinut v roce 2005 pro použití při detekci lidských postav v obraze [25]. Jeho výhodou je robustnost vůči změnám jasu a ostrosti v obraze.

Při výpočtu hodnoty příznakového vektoru HOG se pro každý pixel ve vstupním obrázku vypočítá gradient G_x ve směru osy X tak, že od hodnoty pixelu se souřadnicí [X+1, Y] se odečte hodnota pixelu se souřadnicí [X-1, Y]. Podobným způsobem se vypočítá i gradient G_y ve směru osy Y. Jsou však použity pixely se souřadnicemi [X, Y+1] a [X, Y-1].

Tyto hodnoty jsou následně využity pro výpočet velikosti gradientu $|G|$ a jeho směru ϕ_G. K výpočtu velikosti gradientu se dá použít Pythagorova věta, jelikož si lze tento problém představit jako pravoúhlý trojúhelník, kde G_x a G_y jsou jeho odvěsny. Velikost orientovaného gradientu $|G|$ se tedy dá vypočítat pomocí rovnice 7.

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{7}$$

Jako směr jeho orientace ϕ_G je brán úhel mezi odvěsnou G_x a přeponou. Velikost tohoto úhlu bude nabývat hodnot buď od 0 do 180, nebo od 0 do 360 stupňů a je možné jej vypočítat pomocí rovnice 8.

$$\phi_G = \arctan \frac{G_y}{G_x} \tag{8}$$

Podobně jako u příznaku LBPH, který je popsán v kapitole 3.1, je vstupní obrázek rozdělen na buňky. Pro každou z těchto buněk je vypočítán histogram orientací [26].

Hodnoty sloupců tohoto histogramu se počítají na základě orientace jednotlivých orientovaných histogramů v buňce. Každý sloupec tohoto histogramu reprezentuje určité rozmezí hodnot úhlu ϕ_G. Pokud je toto rozmezí rovno 10 stupňům, bude histogram obsahovat 18 sloupců. Existuje více metod pro výpočet hodnot sloupců histogramu.

- Každý sloupec histogramu obsahuje pouze počet orientovaných gradientů, jejichž orientace ϕ_G se nachází v rozmezí daného sloupce.
- Každý sloupec histogramu obsahuje pouze součet velikostí $|G|$ orientovaných gradientů, jejichž orientace ϕ_G se nachází v rozmezí daného sloupce.
- Každý orientovaný gradient přispívá váženou hodnotou i do sloupců histogramu, které sousedí se sloupcem, do kterého podle svého úhlu spadá.

  Je-li velikost rozmezí sloupce 10 stupňů a $\phi_G = 18$, je část hodoty $|G|$ přičtena do sloupce, kterému náleží rozmezí od 10 do 19 stupňů a část do sloupce, kterému náleží rozmezí od 20 do 29 stupňů.

Následně dochází k normalizaci hodnot histogramů. Toto je provedeno z důvodu dosažení invariance deskriptoru vůči změnám jasu a kontrastu v obraze [24]. Příznakové vektory stejných

objektů se po normalizaci budou méně lišit, přestože jsou snímky, obsahující tyto objekty jinak jasné a ostré [27].

Vstupní obrázek je rozdělen do bloků tak, že každý blok obsahuje 2x2 buňky. Jelikož jsou použity čtvercové bloky, označuje se tento typ jako R-HOG. Dají se také využít bloky kruhové. Takové deskriptory se nazývají C-HOG.

Jednotlivé bloky se vzájemně překrývají. Každá buňka je proto popsána více bloky. Histogramy buněk bloku se zřetězí do jednoho velkého histogramu, který je následně normalizován. Existuje více způsobů, jak může být normalizace dosaženo.

L^1-norm

$$\vec{v}' = \frac{\vec{v}}{(||\vec{v}||_1 + e)} \tag{9}$$

pomocí L^2-norm

L^1-sqrt

$$\vec{v}' = \sqrt{\frac{\vec{v}}{(||\vec{v}||_1 + e)}} \tag{10}$$

L^2-norm

$$\vec{v}' = \frac{\vec{v}}{\sqrt{(||\vec{v}||_2^2 + e^2)}} \tag{11}$$

L^2-hys

Histogram je nejdříve normalizován pomocí L^2-norm a následně dochází k omezení hodnot jeho sloupců na maximální stanovenou hodnotu. Poté je histogram opět normalizován metodou L^2-norm [25].

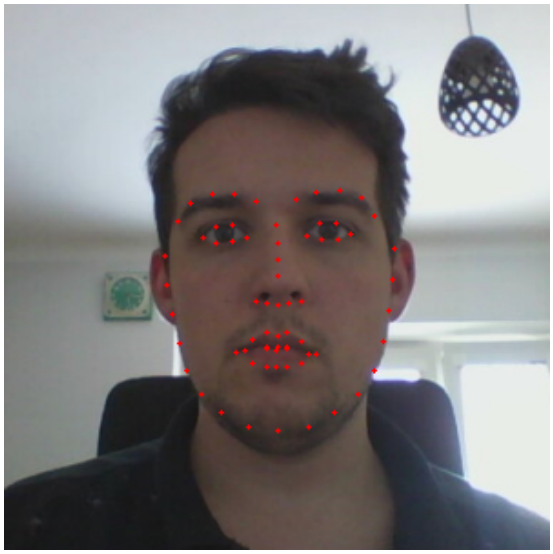
$\vec{v}$ reprezentuje nenormalizovaný vektor obsahující hodnoty histogramu, $\vec{v}'$ je normalizovaný vektor a e je malá pozitivní konstanta, která zabraňuje dělení nulou.

Histogramy všech bloků jsou následně zřetězeny a výsledný histogram je použit jako příznakový vektor.

3.3 Detekce klíčových bodů v obličeji

Řidičův výraz v obličeji se také dá popsat pomocí klíčových bodů v obličeji. Pro tento účel se nejvíce hodí klíčové body nacházející se kolem důležitých částí obličeje, jako jsou rty, oči, obočí či nos. Hodnoty příznakového vektoru mohou tvořit souřadnice těchto bodů v normalizovaném snímku obličeje, nebo úhly mezi nimi.

Pro lokalizaci těchto bodů v obličeji může sloužit například Facemark API, které je dostupné v knihovně OpenCV. Tento extraktor příznaků se však nejdříve musí natrénovat na datasetu,



Obrázek 13: Obrázek obličeje s vyznačenými klíčovými body podle datasetu ibug 300W face dataset

který obsahuje obrázky obličejů s anotací obsahující pozice jejich klíčových bodů. Existuje několik dostupných datasetů, kde již byly tyto body v obličeji vyznačeny.

- Dataset Helen [28], byl vytvořený spoluprací Illinoiské Univerzity v Urbana Champaign, společnosti Adobe Systems Inc. a společnosti Facebook Inc. Tento dataset v každém obličeji vyznačuje 194 bodů okolo obočí, očí, nosu, rtů, a dolní čelisti.
- Ibug 300W face dataset [29, 30, 31] vytvořený na Královské univezitě v Londýně. Stejně jako Helen značí body kolem obočí, očí, nosu, rtů a dolní čelisti. Na rozdíl od datasetu Helen k tomu ale používá 68 bodů.

Výhodou příznakového vektoru tvořeného klíčovovými body v obličeji je jeho invariance v různých světelných podmínkách. Jeho hodnoty budou stejné jak v ostrém tak v rozptýleném světle.

4 Knihovny pro detekci anomálií

Programování všech aspektů detektoru anomálií může být složité a pro programátory s nízkou znalostí lineární algebry velmi náročné. Je však dostupná celá řada otevřených knihoven, které implementaci detektoru anomálií výrazně zjednoduší. Tato kapitola se zabývá těmito knihovnami se zaměřením na knihovny dostupné pro jazyk C++.

4.1 OpenCV

OpenCV (Open Source Computer Vision Library) je open source knihovna, která se zaměřuje na algoritmy pro strojové vidění a strojové učení [32]. Je velice populární díky velkému počtu dostupných nástrojů pro práci s obrazem, multiplatformnosti a dostupnosti pro populární programovací jazyky C++, Python, Java a MATLAB.

Pro účely detekce anomálií v obličeji řidiče se dají využít nástroje pro detekci obličejů v obraze a naimplementované algoritmy pro strojové učení jako je algoritmus k-nejbližších sousedů, SVM, umělé neuronové sítě a hluboké neuronové sítě.

Dále jsou v OpenCV naimplementované i nástroje pro výpočet příznakových deskriptorů SIFT, SURF, které jsou však patentovány a pro použití v komerčních aplikacích musí být licencovány. OpenCV také nabízí API Facemark, které se používá pro vyznačování důležitých bodů v obličeji.

4.2 Dlib

Dlib je otevřená knihovna dostupná pro jazyk C++ a částečně i pro Python [33, 34]. Jejím autorem je Davis King.

Tato knihovna má mnohem širší zaměření, než OpenCV a obshauje například i nástroje pro práci se sockety, GUI Toolkit či threading API.

Obsahuje taky rozsáhlé nástroje pro zpracování obrazu. Dlib nabízí nástroje pro detekci a rozpoznávání obličejů a implementaci příznakových deskriptorů SURF a HOG. Knihovna také obsahuje implementaci některých algoritmů pro strojové učení, jako je SVM či DNN. Při použití této knihovny v oblasti strojového vidění, jsou velkou výhodou implementované funkce pro převod mezi datovým typem Mat knihovny OpenCV a datovými typy knihovny Dlib. Obě knihovny tak mohou být snadno použity společně v jednom projektu.

4.3 TensorFlow

TensorFlow je open source knihovna zaměřená na strojové učení a neuronové sítě. Stabilní API je dostupné pro jazyky Python a C++. Existují ale i verze pro jiné jazyky jako JavaScript, Go nebo Java. Jedná se o velice populární knihovnu, takže existuje mnoho zdrojů, podle kterých se může programátor řídit. Tato knihovna je vyvíjena společností Google.

TensorFlow nabízí několik úrovní abstrakce, takže jej mohou požívat nejen programátoři s rozsáhlými zkušenostmi v oblasti neuronových sítí, ale i nováčci v této oblasti [35].

4.4 Apache MXNet

Apache MXNet je otevřený framework zaměřený na vytváření, trénování a nasazení hlubokých neuronových sítí [36, 37]. Je dostupný pro programovací jazyky C++, Python, R, Java, Scala, Julia, Perl a Clojure. Tento framework se vyznačuje jednoduchou škálovatelností pro trénování neuronové sítě na více GPU či více počítačích a přenositelností natrénovaných sítí, takže neuronová síť natrénovaná na výkonném počítači může být spuštěna na mobilním zařízení. Tento framework je podporovaný cloudovými platformami Microsoft Azure a Amazon WS.

4.5 LIBSVM

LIBSVM je open source knihovna vyvíjená Národní taiwanskou univerzitou. Knihovna je psána v jazyce C++, ale existují i verze pro jazyky Java, R, Matlab a Python. Tato knihovna nabízí nástroje pro práci s SVM a jádrovými funkcemi a implementuje i OC-SVM [38].

5 Implementace detektoru anomálií v obličeji řidiče

5.1 Použité technologie

Pro implemtaci detektoru jsem se rozhodl použít programovací jazyk C++ ve standardu C++11. Pro použití C++ jsem se rozhodl kvůli velkému množství knihoven, které jsou pro tento jazyk dostupné. Také jsem při implementaci použil platformu CUDA, pomocí které jsem akceleroval některé výpočty na grafické kartě. Při implementaci jsem použil následující knihovny:

- **Dlib** pro detekci obličejů a OC-SVM
- **OpenCV** pro jednoduché načítání vstupů z kamer a videí a zobrazování výsledků detekce
- **CUDA Toolkit** pro akceleraci výpočtů na grafické kartě
- **cxxopts** pro jednoduché zpracování vstupů z příkazové řádky
- **libzippp** pro práci se .zip soubory

Projekt byl testovaný pouze na počítači s operačním systémem Ubuntu 19.10.

5.2 Získání řidičova obličeje

Jelikož klasifkátor pracuje pouze s normalizovanými snímky obličeje, je nutné před samotnou klasifikací obličej ve vstupním snímku najít a normalizovat. Toto je provedeno ve dvou krocích. Nejdříve získám snímek z kamery, nebo z video souboru, ve kterém následně detekuji řidičův obličej, který normalizuji a předám detektoru pro klasifikaci.

Kameru snímající řidičův obličej jsem po několika experimentech s její polohou umístil přímo před řidiče na palubní desku. Snímá tak řidičův obličej zepředu z výšky jeho očí. Učinil jsem tak, jelikož použitý detektor obličejů měl při snímání řidiče z velkého úhlu problémy s detekcí. Kamera by však v automobilu mohla být umístěná například i na levém předním sloupku, u vnitřního zpětného zrcátka, nebo na palubní desce vedle volantu.

5.2.1 Získání snímku

Pro získání snímku je použita třída knihovny OpenCV `VideoCapture`. Tato třída nabízí jednoduchý způsob získávání snímků z webkamer i z video souborů různých formátů.

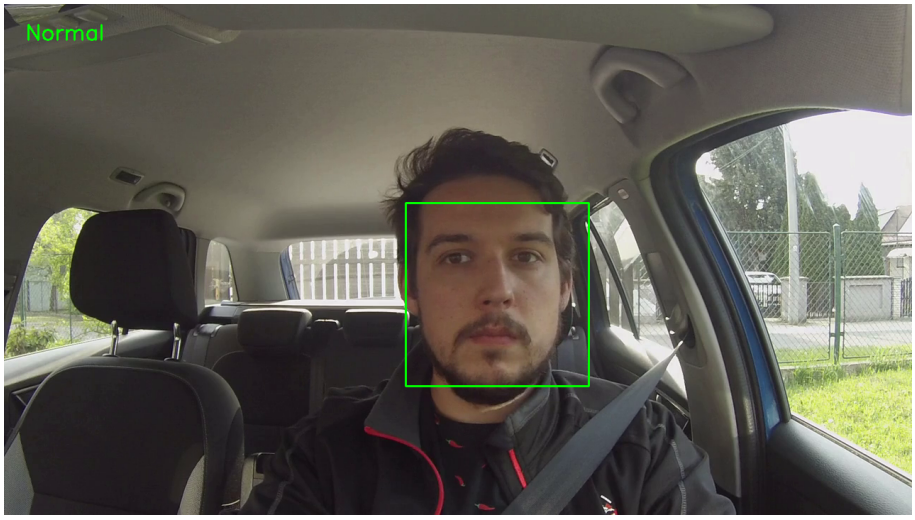
Pro potřeby tohoto programu jsem třídu `VideoCapture` obalil do vlastní třídy `FrameGetter`, která se stará nejen o jednotný způsob získávání snímků ze souborů i webkamer, ale i získání pořadového čísla každého snímku, což je potřebné při výpočtu přesnosti implementovaného detektoru. Tato třída také dovoluje používat více videí, jako by to byl jediný zdroj snímků.

5.2.2 Detekce a normalizace obličeje řidiče

O detekci řidičova obličeje se stará třída `DriverFaceDetector`. V jejím jádru leží detektor `frontal_face_detector` z knihovny Dlib.

Jelikož se ve snímku nemusí nacházet pouze obličej řidiče, ale i obličej spolujezdce či obličeje pasažérů na zadním sedadle, musí z nich třída `DriverFaceDetector` vybrat ten, o kterém si myslí, že patří řidiči. Při výběru řidičova obličeje vycházím z předpokladu, že řidič sedí ke kameře nejbliže. Třída `DriverFaceDetector` proto projde všechny obličeje, které byly ve snímku nalezeny, a jako řidičův zvolí největší z nich.

Výstupem po detekci je objekt třídy `Face`. Tento objekt obsahuje informaci, zda byl ve snímku nějaký obličej nalezen, funkci pro získání oblasti, kde se ve snímku obličej nachází, a funkci pro získání normalizovaného obličeje, který je používán pro detekci anomálií. Při normalizaci je obličej oříznut, převeden do odstínů šedi a jeho velikost je změněna na 96*96 pixelů.



Obrázek 14: Detekovaný obličej řidiče ve snímku

5.3 Sestavení příznakového vektoru

Než je obličej detektorem klasifikován, je z něj vytvořen příznakový vektor. V této práci jsem implementoval dva typy příznakových vektorů.

- Příznakový vektor založený na příznacích HOG, které jsou popsány v kapitole 3.2 na straně 24. Pro extrakci těchto příznaků je použita funkce `extract_fhog_features()`, která je dostupná v knihovně Dlib.
- Příznakový vektor založený na příznacích LBPH, které jsou popsány v kapitole 3.1 na straně 23. Vytváření tohoto příznakového vektoru je akcelerováno pomocí grafické karty. Na té se nejdříve vypočítají LBP hodnoty pro každý pixel obrázku obličeje. Následně se spustí

vlákno pro každý z histogramů, které spočítá četnosti výskytu jednotlivých LBP hodnot v patřičné buňce snímku obličeje.

```
__global__ void calculateHistogram(unsigned char* lbpImg, double*
    histogram, int gridSize, int width, int height) {
    int xFrom = threadIdx.x * gridSize;
    int yFrom = threadIdx.y * gridSize;
    int histStart = 256 * (threadIdx.y * (width / gridSize) + threadIdx.x)
        ;

    for (int i = 0; i < 256; i++) {
        histogram[i + histStart] = 0;
    }

    for (int x = 0; x < gridSize; x++) {
        for (int y = 0; y < gridSize; y++) {
            int nX = x + xFrom;
            int nY = y + yFrom;
            int value = lbpImg[nX+nY*width];
            histogram[value + histStart]++;
        }
    }
}
```

Výpis 1: Výpočet příznaku LBPH pomocí technologie CUDA

U obou těchto typů jsem také měnil velikost buněk a bloků, ze kterých se vypočítávají histogramy, tvořící jednotlivé části příznakového vektoru. Zajímalo mě, jak se změní přesnost a rychlost klasifikace snímku s různými velikostmi těchto buněk. Nechtěl jsem, aby byl výsledný příznakový vektor větší, než by byl vektor tvořený nezpracovanými hodnotami pixelů původního obrázku. Zvolené velikosti buněk jsou vypsány v tabulce 1.

5.4 Detekce anomálií

Rozhodl jsem se, že pro detekci anomálií budu v této práci používat klasifikátor OC-SVM. Princip jeho fungování je popsaný v kapitole 2.1.1. Použil jsem implementaci OC-SVM dostupnou v knihovně Dlib.

Předtím, než může být klasifikátor použit pro detekci, musí být natrénován na normálních datech. Nejdříve deklaruji, jak bude vypadat příznakový vektor a jakou jádrovou funkci bude klasifikátor používat. Pro trénované detektory jsem zvolil radiální bázovou jádrovou funkci.

Typ příznaku	Velikost buňky	velikost příznakového vektoru
HOG	5	8959
HOG	8	3100
HOG	12	1116
HOG	16	496
HOG	24	124
LBPH	16	9216
LBPH	24	4096
LBPH	32	2304

Tabulka 1: Velikosti příznakových vektorů

```
//příznakový vektor HOG s velikostí buňky 8*8 pixelů
typedef dlib::matrix<double, 3100, 1> sampleType;
//jádrová funkce
typedef dlib::radial_basis_kernel<sampleType> kernelType;

//datová struktura obsahují instance normální třídy pro natrénování detektoru
std::vector<sample_type> samples;
```

Výpis 2: Deklarování podoby příznakového vektoru a jádrové funkce

Z trénovacího datasetu jsou získány všechny příznakové vektory, které jsou uloženy do pole. Následně jsou použity pro trénování klasifikátoru. V knihovně Dlib je za toto zodpovědná třída `svm_one_class_trainer`. Poté, co je vytvořen objekt této třídy, je mu předána jádrová funkce, kterou bude klasifikátor používat. U této jádrové funkce se musí nastavit parametr gamma.

Tento parametr ovlivňuje, jak velký vliv mají jednotlivé instance v trénovacím datasetu na pozici nadroviny vložené do příznakového prostoru. Pokud má gamma vysokou hodnotu, mají na pozici nadroviny vliv pouze nejbližší instance. Pokud má gamma nízkou hodnotu, je pozice nadroviny ovlivněna i vzdálenějšími instancemi. Následně je spuštěno trénování klasifikátoru a nadrovina je vložena do příznakového prostoru.

```
dlib::svm_one_class_trainer<kernelType> trainer;
trainer.set_kernel(kernelType(this->kernelWidth));
this->decisionFunction = trainer.train(samples);
```

Výpis 3: Trénování klasifikační funkce

Výslednou funkci, kterou `svm_one_class_trainer` vrátí, lze použít pro klasifikování jednotlivých instancí. Při svém volání požaduje jako parametr příznakový vektor klasifikovaného obličeje.

Výslednou hodnotou této funkce však není binární hodnota, která by říkala, ke které třídě instance náleží. Místo toho tato funkce vrací číselnou hodnotu, která je vysoká, pokud je instance

funkcí klasifikovaná jako normální, a nízká, pokud se jedná o anomálii. Je třeba stanovit práh, který bude oddělovat obě třídy. Porovnáním získané hodnoty s tímto prahem je zjištěna výsledná třída klasifikované instance.

```
bool SVMAnomalyDetectorHOG<TGridSize>::isAnomalous(dlib::array2d<unsigned char>
     &face) {
    auto sample = faceToSampleType(face);
    return this->decisionFunction(sample) < this->threshold;
}
```

Výpis 4: Funkce pro klasifikování obličeje

Po natrénování se výsledný detektor serializuje do souboru typu zip, který obsahuje dva soubory. První z těchto souborů obsahuje serializovanou funkci získanou pomocí třídy `svm_one_class_trainer`. Druhý ze souborů obsahuje metadata o detektoru, která jsou potřebná pro instancování správného objektu detektoru. Nachází se zde informace, jakou metodu redukce dimenze a velikost buňky detektor používá pro sestavení příznakového vektoru a také hodnota proměnné `threshold`, která je používaná při detekci. Uživatel tak v tomto souboru může změnit velikost této proměnné.

(a) Obličej klasifikovaný jako anomální (b) Obličej klasifikovaný jako normální

Obrázek 15: Příklad klasifikovaných obličejů

6 Testování implementovaného detektoru

Tato kapitola se věnuje testování detektoru implementovaného v kapitole 5. Detektor byl testovaný na počítači s procesorem Intel Core i7-4710MQ, 16GB RAM a grafickou kartou Nvidia Quadro K610M. Aby mohl být detektor nasazen v reálném provozu, jsou od něj požadovány následující dvě vlastnosti:

1. Detektor musí být schopen klasifikovat snímek řidiče v dostatečně rychlém čase, aby vozidlo dokázalo včas na případnou anomálii zareagovat. Klasifikace snímku by neměla trvat déle než pár desetin sekundy, jinak by už reakce na vzniklou anomálii mohla přijít příliš pozdě. V ideálním případě by detektor měl zvládat klasifikaci dostatečně rychle na to, aby stíhal klasifikovat všechny snímky ve vstupním video streamu. U videa se snímkovou frekvencí 30 snímků za sekundu by tedy detektor měl být schopen zpracovat a klasifikovat snímek za méně než 0,2 sekundy.

2. Klasifikace implementovaného detektoru musí být dostatečně přesná. Je důležité, aby detektor neklasifikoval normální situace jako anomální a naopak. Pokud by systém vytvářel falešné poplachy, pravděpodobně by u řidiče nesklidil velký úspěch a ten by jej nejspíše vypnul. V případě, že by systém anomálie nedetekoval, by nesplnil svůj účel.

Při testování mě zajímá, jaký vliv má zvolený způsob redukce dimenzionality a velikost výsledného příznakového vektoru na výkon a přesnost detektoru. Předpokládám, že detektory používající menší příznakové vektory budou při detekci anomálií přesnější než detektory s většími příznakovými vektory, protože podle mě budou méně podléhat prokletí dimenzionality. Detektory s většími příznakovými vektory by však mohly fungovat lépe ve chvíli, kdy se anomální situace projevuje hlavně jemnou mimikou obličeje, jelikož velikost buněk, ze kterých se příznakový vektor vytváří je u nich menší a přesněji tak popisují, co se v obličeji děje.

6.1 Metodika testování

Tato kapitola popisuje, jakým způsobem jsem testoval výše uvedené vlastnosti implementovaných detektorů anomálií.

Pro jejich testování jsem použil dvě videa. V prvním videu je natočený řidič během normálního řízení automobilu. Ve videu se plně věnuje řízení, dívá se před sebe na vozovku, do zrcátek a na přístrojovou desku. Toto video je použito pro natrénování testovaných detektorů.

Druhé video obsahuje kombinaci normálních i anomálních situací. Kromě činností, které jsou obsaženy v trénovacím videu, v něm řidič používá mobilní telefon, předstírá únavu a spánek, ladí rádio, pije z lahve, kýchá, mhouří oči a zívá. Toto video je použito pro vypočítání přesnosti detektoru. Pozice těchto anomálních situací v tomto videu jsou sepsány v separátním souboru a jsou používány pro zjištění, zda je situace detektorem správně klasifikována.

6.1.1 Rychlost detekce

U měření rychlosti detekce nejde pouze o rychlost samotné klasifikace, ale o rychlost celého zpracování snímku. Měřím tedy i dobu, po kterou trvá detekce obličeje ve snímku a následné vytvoření příznakového vektoru. Měření doby klasifikace začínám ve chvíli, kdy se načte snímek do paměti a ukončím jej, jakmile získám od klasifikátoru verdikt, zda se řidič tváří anomálně.

Abych dostal dostatečně vypovídající výsledek, provedu toto měření pro všechny snímky v testovacím videu, ve kterých detetor obličejů našel tvář řidiče. Ze získaných hodnot vypočítám minimální, maximální a střední hodnotu s průměrnou a relativní odchylkou. Tyto hodnoty budou výsledkem mého měření.

6.1.2 Přesnost detektoru

Při měření přesnosti detektoru se klasifikují všechny snímky v testovacím videu a výsledné třídy se porovnají s hodnotami obsaženými v přiloženém souboru s informacemi, kde se v testovacím videu anomálie nacházejí. Z výsledků těchto porovnání je následně sestavena matice záměn. Tato matice má 4 buňky.

Pravdivě pozitivní (PP) Tato buňka obsahuje počet snímků, na kterých byl výraz řidiče správně klasifikován jako normální.

Falešně pozitivní (FP) Tato buňka obsahuje počet snímků, na kterých byl výraz řidiče klasifikován jako normální, i když ve skutečnosti byl anomální.

Pravdivě negativní (PN) Tato buňka obsahuje počet snímků, na kterých byl výraz řidiče správně klasifikován jako anomální.

Falešně pozitivní (FN) Tato buňka obsahuje počet snímků, na kterých byl výraz řidiče klasifikován jako anomální, i když ve skutečnosti byl normální.

		skutečná třída	
		Normální	**Anomální**
klasifikovaná třída	**Normální**	Pravdivě pozitivní	Falešně pozitivní
	Anomální	Falešně negativní	Pravdivě negativní

Tabulka 2: Matice záměn

Z hodnot buněk výsledné matice záměn se následně dá vypočítat přesnost *(ACC)* testovaného detektoru pomocí rovnice 12.

$$ACC = \frac{PP + PN}{PP + PN + FP + FN} \tag{12}$$

Dále mě u detektorů zajímají jejich pozitivní a negativní prediktivní hodnoty. Z těchto hodnot se dá zjistit, jestli častěji dochází k chybám v klasifikaci normálních hodnot, nebo při klasifikaci anomálií.

Negativní prediktivní hodnota *(NPV)* udává pravděpodobnost, že instance klasifikovaná jako anomální je skutečně anomální. Tato hodnota bude nabývat nižších hodnot v případě, že detektor často klasifikuje normální situace jako anomální. Hodnota NPV se dá vypočítat pomocí rovnice 13.

$$NPV = \frac{PN}{PN + FN} \tag{13}$$

Pozitivní prediktivní hodnota *(PPV)* udává pravěpodobnost, že instance klasifikovaná jako normální je skutečně normální. Tato hodnota bude nabývat nižších hodnot v případě, že detektor často klasifikuje anomální situace jako normální. Hodnota PPV se dá vypočítat pomocí rovnice 14.

$$PPV = \frac{PP}{PP + FP} \tag{14}$$

6.2 Výsledky testování

V tabulce 3 jsou vypsány hodnoty, které byly použity pro natrénování testovaných detektorů. Hodnoty proměnných `gamma` a `threshold` byly zadány experimentálním způsobem tak, aby hodnota přesnosti detekce byla co nejvyšší. U těchto detektorů srovnávám jejich vlastnosti popsané v kapitole 6.1.

detektor	**gamma**	**threshold**
HOG5	0,001	0,05
HOG8	0,001	0
HOG12	0,001	0,05
HOG16	0,001	0,05
HOG24	0,001	0
LBPH16	0,000 000 1	-0,1
LBPH24	0,000 000 1	-0,01
LBPH32	0,000 000 1	-0,01

Tabulka 3: Použité hodnoty testovaných detektorů

6.2.1 Rychlost detekce

Tabulka 4 obsahuje naměřené hodnoty rychlosti klasifikace obličeje. Z těchto hodnot je vidět, že detektory používající rozsáhlejší příznakové vektory potřebují pro klasifikaci snímku více času.

Tento časový rozdíl je podle mě zaviněn kopírováním hodnot při sestavování příznakového vektoru. U příznaků LBPH se totiž musí hodnoty zkopírovat z paměti grafické karty, na které je jejich výpočet akcelerován, do paměti příznakového vektoru. V případě příznaků HOG se zase jeho hodnoty kopírují z třídimenzionální datové struktury do jednodimenzionálního vektoru. Jelikož detektory s většími příznakovými vektory musí kopírovat více hodnot, jsou pomalejší.

detektor	**průměrný čas [μs]**	**rel. odchylka [%]**	**nejkratší čas [μs]**	**nejdelší čas [μs]**
HOG5	7320,84 ± 67.43	0.92	7234	9025
HOG8	2692.26 ± 32.78	1.22	2648	4520
HOG12	1213.56 ± 95.10	7.83	1114	2316
HOG16	679.16 ± 10.65	12.97	644	868
HOG24	405.26 ± 21.82	5.38	376	816
LBPH16	5247,65 ± 225,4	4,3	4917	64350
LBPH24	3639,49 ± 491.07	13.49	3122	57882
LBPH32	3037,7 ± 393.97	12.97	2587	54169

Tabulka 4: Naměřené hodnoty rychlosti klasifikace

Doba zpracování snímku se u všech testovaných detektorů pohybuje nejvýše v jednotkách milisekund. Pro nasazení v reálném provozu by tedy všechny z nich byly dostatečně rychlé a dokázaly by bez problémů klasifikovat snímky z kamery.

6.2.2 Přesnost detekce

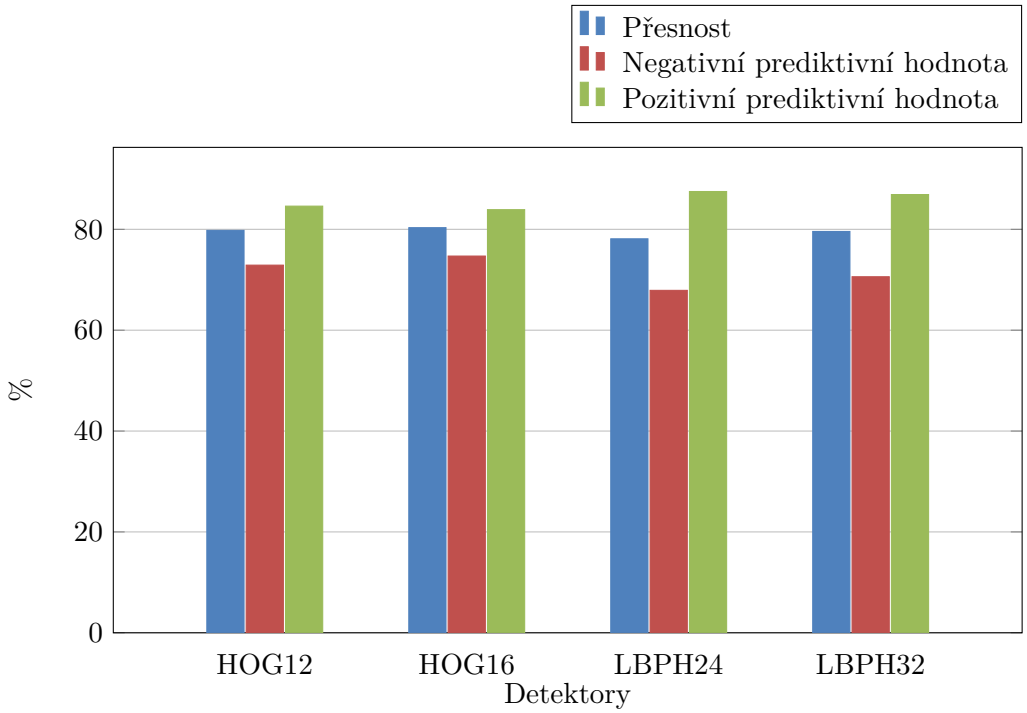
Z tabulky 5, která obsahuje výsledné hodnoty přesností a negativních a pozitivních prediktivních hodnot testovaných detektorů, je patrné, že můj předpoklad z kapitoly 6 je správný. Detektory využívající při detekci menší příznakové vektory byly skutečně při detekci přesnější a méně podléhaly prokletí dimenzionality, než detektory pracující s většími příznakovými vektory.

U detektoru HOG24 je ale vidět, že pokud je dimenze příznakového vektoru příliš zmenšená, neobsahuje už příznakový vektor dostatek informací pro správnou klasifikaci a přesnost detektoru klesá.

detektor	**ACC [%]**	**NPV [%]**	**PPV [%]**
HOG5	74.39	65.5	81,1
HOG8	76.27	69,0	81,1
HOG12	79.79	72,9	84,6
HOG16	80,33	74,7	83,9
HOG24	77.66	77.8	77.6
LBPH16	76.87	72,1	79,6
LBPH24	78,12	67,9	87,5
LBPH32	79.58	70.6	86,9

Tabulka 5: Hodnoty přesnosti (ACC), negativní prediktivní hodnoty (NPV) a pozitivní prediktivní hodnoty (PPV) testovaných detektorů

Velikost příznakového vektoru však není jediný faktor, který měl vliv na výsledný výkon detektorů. Svou roli sehrála i zvolená metoda redukce dimenzionality. Srovnání nejlepších detektorů využívajících příznaky HOG a LBPH je zobrazeno v následujícím grafu.



Obrázek 16: Srovnání detektorů HOG12, HOG16, LBPH24 a LBPH32

Jak můžeme v naměřených hodnotách vidět, byla přesnost detektorů LBPH32 a LBPH24 srovnatelná s HOG16 a HOG12, přestože jejich příznakové vektory byly mnohem větší.

Detektory používající příznaky LBPH měly vyšší pozitivní prediktivní hodnotu, než detektory používající příznaky HOG, které ale naopak měly vyšší negativní prediktivní hodnotu. Právě kvůli jejich vyšší negativní prediktivní hodnotě, považuji detektory HOG12 a HOG16 za lepší, než detektory LBPH24 a LBPH32. Myslím si totiž, že je důležité, aby detektor jako anomální klasifikoval jenom situace, které opravdu anomální jsou a neobtěžoval řidiče v normálních situacích.

7 Možné zlepšení přesnosti detekce pomocí dalších senzorů

Mnou navržené detektory anomálií klasifikují situaci pouze na základě snímků pořízených kamerou umístěnou v kabině automobilu. Existuje však řada dalších hodnot, které by se daly pro detekci anomálií použít a jejich přidání do příznakového vektoru implementovaného detektoru by mohlo zlepšit přesnost detekce anomálií. Pro tento účel se dají použít fyziologické informace o řidiči a informace získané z automobilu.

Přidané informace pomohou přesnosti detekce hlavně v případě, kdy se detektor potýká s kontextovými anomáliemi. Tím, že příznakový vektor bude obsahovat i kontext pro tyto anomálie, stanou se z nich anomálie bodové a jejich klasifikace již nebude tak problémová. Určení kontextu však může být v některých situacích velmi složité. Kontextové i bodové anomálie jsou vysvětleny v kapitole 2 na straně 13. Příznakový vektor by mohl obsahovat například následující informace.

Rychlost automobilu

V situaci, kdy je ze vstupního snímku patrné, že se řidič nedívá před sebe, ale mimo směr jízdy vozidla, nemusí detektor z pouhých obrazových dat poznat, zda je to z důvodu, že se řidič nevěnuje řízení. Ve chvíli, kdy se řidič dívá mimo směr jízdy při vysoké rychlosti by bylo vhodné řidiče upozornit, že by se měl věnovat řízení. Pokud tak však činí při nízkých rychlostech, není to tak nebezpečné a je pravděpodobné, že pro to má dobrý důvod, jako je rozhlížení se před vjezdem do křižovatky.

Pokud by příznakový vektor obsahoval informaci, jakou rychlostí řidič jede, získal by detektor kontext o tom, v jaké situaci se řidič nachází, a bylo by pro něj mnohem jednodušší tuto situaci klasifikovat.

Intenzita světla v automobilu

Tato informace by mohla být nápomocná například v situaci, kdy má řidič přivřené oči. Jede-li řidič proti slunci, které mu svítí do očí, je taková situace nejspíše v pořádku. Je-li to však z důvodu únavy, měla by tato situace být označena jako anomální a řidič by měl být upozorněn.

Úhel natočení volantu

Význam této informace je podobný, jako rychlost automobilu. Pokud řidič zatáčí, měl by se tomuto manévru plně věnovat a dívat se směrem, kterým jede. Hodnota úhlu natočení volantu se běžně požívá v komerčeně dostupných systémech, jako je Driver drowsiness detection od firmy BOSCH [39] nebo Rest recomendation system od výrobce automobilů Audi [40].

Pozice automobilu mezi podélným značením

Stejně jako úhel natočení volantu, je i informace o pozici automomobilu mezi podélným značením využívána komerčně dostupnými systémy. Je zřejmé, že vyjíždí-li řidič z pruhu,

aniž by to byl jeho záměr, nevěnuje se plně řízení a situace by měla být klasifikována jako anomální.

Hodnota PERCLOS

Hodnota PERCLOS udává, jakou část předem stanoveného časového úseku měl řidič zavřené oči. Ve snímku z kamery umístěné v automobilu se detekují řidičovy oči a pokud jsou přivřené alespoň z 80 procent, bere je algoritmus jako zavřené [41]. Výsledná procentuální hodnota může být použita pro klasifikování, zda je řidič unavený, jelikož se unavenému řidiči častěji zavírají oči. Nevýhodou je, že tato hodnota může být ovlivněna vnějšími faktory, jako je ostré světlo ze slunce, nebo z protijedoucích automobilů.

Fyziologické hodnoty

Pro dřívější detekování únavy jsou fyziologické hodnoty vhodnější, než hodnoty získané ze senzorů automobilu, jelikož se v nich řidičova únava projevuje mnohem dříve [7]. Pro detekování anomálií se například dají využít hodnoty elektroencefalografu (EEG) měřícího mozkovou aktivitu, elektrookulografu (EOG) měřícího aktivitu očí či elektromyografu (EMG) měřícího elektrické biosignály vycházející ze svalů. Získání těchto hodnot, aniž by to nějak omezovalo nebo obtěžovalo řidiče, je však složité. Nejjednodušší je získání hodnot tepové frekvence a tělesné teploty z čidel umístěných ve volantu, nebo v sedačce.

8 Závěr

Technologie kontrolující pozornost řidiče během jízdy se v současnosti stávají běžnou výbavou automobilů. Nepozornost řidiče je totiž jednou z nejčastějších příčin autonehod, jelikož se řidiči často při řízení věnují jiným činostem, jako je kontrola sociálních sítí nebo nastavování navigace. Systémy sledující řidiče jej v případě, že detekují pokles jeho pozornosti, upozorní, aby se řízení plně věnoval, nebo aby zastavil a dal si krátkou přestávku. Takovéto systémy by mohly zachránit mnoho lidských životů.

Cílem této bakalářské práce bylo seznámit se s metodami, které je možné využít pro detekci anomálií v řidičově chování, a navrhnout vlastní detektor, který by tyto anomálie detekoval. Vytvořil jsem několik detektorů používajících algoritmus OC-SVM a lišících se použitými příznakovými vektory.

Za nejlepší z vytvořených detektorů považuji detektor používající příznaky HOG s bloky o velikosti 16 pixelů, který při testování dosáhl přesnosti 80,33%. Svou přesností byl tento detektor sice srovnatelný s detektory používajícími příznaky LBPH, ale měl oproti nim při testování vyšší negativní prediktivní hodnotu. Jeho pozitivní prediktivní hodnota ale naopak byla nižší, než u detektorů používající příznaky LBPH.

To znamená, že tento detektor sice při testování klasifikoval více anomálních situací, jako normální, ale pokud situaci klasifikoval jako anomální, byla vyšší pravděpodobnost, že má pravdu. To je podle mě lepší, než u detektorů používajících příznaky LBPH. Ty totiž při stejné hodnotě přesnosti detektoru klasifikovaly více normálních situací, jako anomální. Řidiče by tedy častěji obtěžovaly v normálních situacích a proto je větší pravděpodobnost, že by je řidič nepoužíval. Detektor, který sice klasifikuje některé anomální situace jako normální, ale neobtěžuje řidiče falešnými poplachy, je rozhodně lepší, než detektor, který je vypnutý.

Přesto si ale nemyslím, že by byl tento detektor dostatečně přesný na nasazení v reálném provozu. Proto jsem se dále v této práci věnoval dalším informacím, které by mohl detektor vyhodnocovat a zlepšit tak přesnost detekce anomálií. Pro tento účel by se dala využít fyziologická data, jako je řidičův tep, nebo data získaná z automobilu, jako například pozice auta mezi podélným značením nebo rychlost jízdy.

Literatura

1. AFFECTIVA. *3 Ways Driver Monitoring Is Improving Road Safety*. 2018-10. Dostupné také z: https://blog.affectiva.com/3-ways-driver-monitoring-is-improving-road-safety.
2. CHANDOLA, VARUN; BANERJEE, ARINDAM; KUMAR, VIPIN. Anomaly Detection: A Survey. *ACM Computing Surveys*. 2009-07. Dostupné z DOI: 10.1145/1541880.1541882.
3. STRAKA, Jan; FABIÁNOVÁ, Jana. *Ročenka nehodovosti na pozemních komunikacích v České republice v roce 2018*. 2019-06. Dostupné také z: https://www.policie.cz/clanek/statistika-nehodovosti-900835.aspx?q=Y2hudW09Mw%3d%3d.
4. VOLVO. *Driver Alert Control*. 2020-01. Dostupné také z: https://www.volvocars.com/cz/support/manuals/s60/2020-early/podpora-ridice/driver-alert-control/driver-alert-control.
5. VOLKSWAGEN. *Driver Alert System*. Dostupné také z: https://www.volkswagen.co.uk/technology/car-safety/driver-alert-system.
6. *Dokáže řidiče varovat před nadměrnou únavou*. Dostupné také z: https://www.mercedes-benz.cz/vans/cs/vito/mixto/optional-equipment-highlights/teaser-group-1/attention-assist.
7. SAHAYADHAS, Arun; SUNDARAJ, Kenneth; LIN, Murugappan Murugappan. Detecting Driver Drowsiness Based on Sensors: A Review. *Sensors (Basel)*. 2012-12. Dostupné z DOI: 10.3390/s121216937. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
8. CHELLAPPA, Y.; JOSHI, N. N.; BHARADWAJ, V. Driver fatigue detection system. In: *2016 IEEE International Conference on Signal and Image Processing (ICSIP)*. 2016, s. 655–660.
9. AMER, Mennatallah; GOLDSTEIN, Markus; ABDENNADHER, Slim. Enhancing One-class Support Vector Machines for Unsupervised Anomaly Detection. *ODD '13: Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*. 2013-08, s. 8–15. Dostupné z DOI: 10.1145/2500853.2500857.
10. WINSTON, Patrick. *16. Learning: Support Vector Machines*. 2014-01. Dostupné také z: https://www.youtube.com/watch?v=_PwhiWxHK8o.
11. KAVZOGLU, T.; COLKESEN, I. A kernel functions analysis for support vector machines for land cover classification. *International Journal of Applied Earth Observation and Geoinformation*. 2009-06, s. 352–359. Dostupné z DOI: 10.1016/j.jag.2009.06.002.

12. SCHÖLKOPF, Bernhard; WILLIAMSON, Robert; SMOLA, Alex; SHAWE-TAYLOR, John; PLATT, John. Support Vector Method for Novelty Detection. In: 1999-01, sv. 12, s. 582–588.

13. MANEVITZ, Larry M.; YOUSEF, Malik. One-Class SVMs for Document Classification. *Journal of Machine Learning Research*. 2001-12, s. 139–154.

14. VLASVELD, Roemer. *Introduction to One-class Support Vector Machines*. 2013-07. Dostupné také z: `http://rvlasveld.github.io/blog/2013/07/12/introduction-to-one-class-support-vector-machines/`.

15. SRIVASTAVA, Tavish. *Introduction to k-Nearest Neighbors: A powerful Machine Learning Algorithm (with implementation in Python & R)*. 2018-10. Dostupné také z: `https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/`.

16. MUNROE, Daniel; MADDEN, Michael. Multi-class and single-class classification approaches to vehicle model recognition from images. 2020-05.

17. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

18. PERERA, Pramuditha; PATEL, Vishal M. Learning Deep Features for One-Class Classification. *IEEE Transactions on Image Processing*. 2019-11, roč. 28, č. 11, s. 5450–5463. Dostupné z DOI: `10.1109/TIP.2019.2917862`.

19. OZA, Poojan; PATEL, Vishal M. One-Class Convolutional Neural Network. *IEEE Signal Processing Letters*. 2019-02, roč. 26, č. 2, s. 277–281. Dostupné z DOI: `10.1109/LSP.2018.2889273`.

20. CHALAPATHY, Raghavendra; KRISHNA MENON, Aditya; CHAWLA, Sanjay. Anomaly Detection using One-Class Neural Networks. *arXiv e-prints*. 2018-02, s. arXiv:1802.06360. Dostupné z arXiv: `1802.06360 [cs.LG]`.

21. KALINA, Jan; TEBBENS, Jurjen Duinter. METODY PRO REDUKCI DIMENZE V MNOHOROZMĚRNE STATISTICE A JEJICH VÝPOČET. Dostupné také z: `http://www.cs.cas.cz/duintjertebbens/pubs/KalinaDT.pdf`.

22. OJALA, Timo; PIETIKHENL, Matti; HARWOOD, David. Performance Evaluation of Texture Measures with Classification Based on Kullback Discrimination of Distributions.

23. COMPUTERPHILE; VALSTAR, Michel. *Faces & the Local Binary Pattern - Computerphile*. 2015. Dostupné také z: `https://www.youtube.com/watch?v=wpAwdsubl1w`.

24. HRÚZ, Marek. *LBP, HoG*. 2015-10. Dostupné také z: `http://www.kky.zcu.cz/uploads/courses/mpv/05/materialy05.pdf`.

25. DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. 2005, sv. 1, 886–893 vol. 1.

26. SINGH, Aishwarya. *Feature Engineering for Images: A Valuable Introduction to the HOG Feature Descriptor*. 2019-09. Dostupné také z: `https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/`.

27. TOMASI, Carlo. *Histograms of Oriented Gradients*. 2017-09. Dostupné také z: `http://db.cs.duke.edu/courses/compsci527/fall17/notes/hog.pdf`.

28. LE, Vuong; BRANDT, Jonathan; LIN, Zhe; BOURDEV, Lubomir; HUANG, Thomas S. Interactive Facial Feature Localization. *European cCnference on Computer Vision (ECCV)*. 2012.

29. SAGONAS, C.; ANTONAKOS, E.; TZIMIROPOULOS, G.; ZAFEIRIOU, S.; PANTIC, M. 300 faces In-the-wild challenge: Database and results. *Image and Vision Computing (IMAVIS)*. 2016, s. 3–18.

30. SAGONAS, C.; TZIMIROPOULOS, G.; ZAFEIRIOU, S.; PANTIC, M. 300 Faces in-the-Wild Challenge: The first facial landmark localization Challenge. In: *Proceedings of IEEE Int'l Conf. on Computer Vision (ICCV-W), 300 Faces in-the-Wild Challenge (300-W)*. Sydney, Australia, 2013-12.

31. SAGONAS, C.; TZIMIROPOULOS, G.; ZAFEIRIOU, S.; PANTIC, M. *Proceedings of IEEE Int'l Conf. Computer Vision and Pattern Recognition (CVPR-W), 5th Workshop on Analysis and Modeling of Faces and Gestures (AMFG 2013)*. A semi-automatic methodology for facial landmark annotation. Oregon, USA, 2013-06.

32. *OpenCV, About*. Dostupné také z: `https://opencv.org/about/`.

33. *Dlib C++ Library*. Dostupné také z: `http://dlib.net/`.

34. *Dlib C++ Library, Introduction*. Dostupné také z: `http://dlib.net/intro.html`.

35. *TensorFlow, Why TensorFlow*. Dostupné také z: `https://www.tensorflow.org/about`.

36. GUPTA, Megha. *MXNet: What is it and How to Get Started*. 2017-06. Dostupné také z: `https://becominghuman.ai/mxnet-what-is-it-and-how-to-get-started-52efd9cb52cf`.

37. *MXNet, Features*. Dostupné také z: `https://mxnet.apache.org/features`.

38. CHANG, Chih-Chung; LIN, Chih-Jen. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*. 2011, roč. 2, s. 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

39. BOSCH. *Bosch Driver Drowsiness Detection*. Dostupné také z: `https://www.bosch-presse.de/pressportal/de/en/bosch-driver-drowsiness-detection-41616.html`.

40. AUDI. *Driver assistance systems*. Dostupné také z: https://www.audi-mediacenter.com/en/foray-into-the-worlds-largest-market-segment-the-audi-a3-sedan-and-s3-sedan-3249/driver-assistance-systems-3350.

41. YAN, J.; KUO, H.; LIN, Y.; LIAO, T. Real-Time Driver Drowsiness Detection System Based on PERCLOS and Grayscale Image Processing. In: *2016 International Symposium on Computer, Consumer and Control (IS3C)*. 2016, s. 243–246.

A Příloha v IS EDISON

Přiložený archiv obsahuje:

zdrojové kódy - adresář obsahující zdrojové kódy bakalářské práce

- **spusteni.pdf** - pdf soubor obsahující návod k instalaci a spuštění přiložených zdrojových kódů
- **CMakeLists.txt** - soubor potřebný pro buildování projektu
- **src** - adresář se zdrojovými kódy bakalářské práce
- **lib** - adresář obsahující přenosné knihovny cxxopts a Dlib

detektory - adresář obsahující natrénované detektory

videa - adresář obsahující videa použitá pro trénování a testování detektorů

- **normal.mp4** - video používané pro trénování detektorů
- **anomalies.mp4** - video používané pro testování detektorů
- **anomalies.csv** - soubor popisující, kde se ve videu anomalies.mp4 nachází anomálie

ukázka_detekce.mp4 - video ukazující použití detektoru HOG16.det na testovací video